"Express Mail" mailing label number:

ET891547952US

AUTOMATIC COMMUNICATION AND SECURITY RECONFIGURATION FOR REMOTE SERVICES

Michael J. Wookey
Trevor Watson
Jean Chouanard

Cross Reference to Related Applications

This application relates to co-pending United States patent application Serial No. _____, attorney docket number P7223, filed on a even date herewith, entitled "Remote Services System Management Interface" and naming Michael J. Wookey, Trevor Watson and Jean Chouanard as inventors, the application being incorporated herein by reference in its entirety.

This application relates to co-pending United States patent application Serial No. _____, attorney docket number P7225, filed on a even date herewith, entitled "Remote Services Message System to Support Redundancy of Data Flow" and naming Michael J. Wookey, Trevor Watson and Jean Chouanard as inventors, the application being incorporated herein by reference in its entirety.

This application relates to co-pending United States patent application Serial No. _____, attorney docket number P7229, filed on a even date herewith, entitled "Remote Services Delivery Architecture" and naming Michael J. Wookey, Trevor Watson and Jean Chouanard as inventors, the application being incorporated herein by reference in its entirety.

This application relates to co-pending United States patent application Serial No. _____, attorney docket number P7230, filed on a even date herewith, entitled "Prioritization of Remote Services Messages Within a Low Bandwidth Environment"

and naming Michael J. Wookey, Trevor Watson and Jean Chouanard as inventors, the application being incorporated herein by reference in its entirety.

This application relates to co-pending United States patent application Serial No. _____, attorney docket number P7231, filed on a even date herewith, entitled "Remote Services System Back-Channel Multicasting" and naming Michael J. Wookey, Trevor Watson and Jean Chouanard as inventors, the application being incorporated herein by reference in its entirety.

This application relates to co-pending United States patent application Serial No. _____, attorney docket number P7233, filed on a even date herewith, entitled "Remote Services System Data Delivery Mechanism" and naming Michael J. Wookey, Trevor Watson and Jean Chouanard as inventors, the application being incorporated herein by reference in its entirety.

This application relates to co-pending United States patent application Serial No. _____, attorney docket number P7234, filed on a even date herewith, entitled "Remote Services WAN Connection Identity Anti-spoofing Control" and naming Michael J. Wookey, Trevor Watson and Jean Chouanard as inventors, the application being incorporated herein by reference in its entirety.

Field of the Invention

The present invention relates to remote service delivery for computer networks and, more particularly, to a method and apparatus for automatically reconfiguring system components upon detection of a communication error.

Background of the Invention

It is known to provide a variety of services that are delivered remotely to a customer. These services range from point solutions delivering specific service to more complex remote service instantiations supporting multiple services. The technology behind these services has a number of things in common: they are

generally a good idea; they provide a valuable service to a set of customers; and, they are generally isolated from one another.

The number of remote services available show the need and demand for such services. However, the fragmentation of the services reduces the overall benefit to the service provider as well as to the customer. The customer is presented with an often confusing issue of which services to use, why the services are different and why the service provider cannot provide a single integrated service.

In the remote services system, certain communication parameters should be available to the various system components to enable the distributed components to establish connection with the network. Examples of such parameters include network parameters (e.g. IP addresses or name of the next hop), network protocol parameters (e.g. the use of a specific HTTP proxy for SLL/HTTP connection or the IP address for a valid mail relay for SMTP connection) or security related parameters (e.g. type of encryption or the identity or certificate to be used).

While the network remote services system provides a way to update all of these configuration files, it should be able to communicate with each of the components. When a communication error is detected the system should be able to diagnose the type of error and immediately reestablish communication with the affected components. There is a need, therefore, for a process by which a remote services system can provide for automatic communication and security reconfiguration to ensure that the system is able to maintain integrity of connectivity and communication with the various system components.

Summary of the Invention

The automatic communication and security reconfiguration system provides a process by which the remote services system can reconfigure system components when communication back to the system servers has been disrupted. If the system detects a communication error related to an identify problem, such as an invalid client

certificate, the system communication module will try to fetch a new certificate from a secure URL pointing to the service provider web site containing data parameters relating to the system. Customer and component information such as the customer ID and the component ID are associated with this request via a HTTP POST. This request is redirected to a web server local to the customer and will reply with the new certificate which is installed automatically.

If the system detects a communication problem not related to identify (cannot reach the MLM, for example), it will then try to fetch its configuration from a secure URL pointing to the service provider web site. Certificate information, such as the customer ID and the component ID, are associated with this request via a HTTP POST. However, this request may be served directly by service provider servers or redirected to be served by a web server local to the customer network.

When a component receives its new configuration under either of the scenarios discussed above, from a local web server or from the service provider, the component will install the configuration and validate the connectivity back to the system servers.

Brief Description of the Drawings

The present invention may be understood, and its numerous objects, features, and advantages made apparent to those skilled in the art by referencing the accompanying drawings.

Figure 1 shows a block diagram of a remote service delivery architecture.

Figure 2 shows a schematic block diagram of the components relating to the remote services infrastructure.

Figure 3 shows a publish and subscribe example using the remote services delivery architecture.

Figure 4 shows a block diagram of the application program interfaces (API's) of the remote service delivery architecture.

Figures 5A and 5B show a more detailed version of the components of Figure 2.

Figure 6 shows a block diagram of a remote services proxy and a remote services system management integrator.

Figure 7 shows a block diagram of a remoter services intermediate mid level manager (MLM).

Figure 8 shows a block diagram of a remote services applications MLM.

Figure 9 shows a block diagram of an application server module.

Figure 10 shows a block diagram of a content generation MLM module.

Figure 11 shows a flow diagram of a remote services system communication.

Figure 12 shows a block diagram of the data blocks that comprise the data that flows through the remote services infrastructure.

Figures 13A and 13B show an example of the high level architecture component relationships of a remote services system that is configured according to the remote services architecture.

Figure 14 is a flowchart illustrating the processing steps for detecting a communication error and for automatically reconfiguring components of the remote services system.

Figure 15 is a flowchart illustrating the processing steps for one of the sub-processes for detecting an identity error and for automatically reconfiguring components of the remote services system.

Figure 16 is a flowchart illustrating the processing steps for one of the sub-processes for detecting a connectivity error and for automatically reconfiguring components of the remote services system.

Detailed Description

Figure 1 shows a block diagram of an architecture for a remote service delivery system 100 that meets the needs of both the service provider and the customer. The architecture of the present invention is modularized to provide broad support for both the customer and the service provider in terms of evolution of service functionality to the architecture and within the architecture.

The architecture is broadly comprised of the remote service infrastructure 102, a group of service modules 103 and a plurality of communications modules 110. The

remote services infrastructure 102 provides reliable remote service delivery and data management. The remote services infrastructure 102 supports the needs of a service creator by focusing the service creator on the needs and the design of the service by eliminating the need for the service creator to be concerned about how data is transferred and managed to and from a customer site.

The remote services infrastructure 102 provides an interface to support the development of services that use a set of common service parameters to develop customized services for a specific service provider or customer. The infrastructure 102 is separately segmented from, but actively interacts with, the service modules 103.

Within the group of software modules 103 are individual software modules that analyze data collected by the remote services infrastructure 102 and provides service value based on that data to a customer. Thus, the remote services infrastructure 102 and the service modules 103 can be differentiated as follows: the remote services infrastructure 102 is concerned with how data is collected, while the service module 103 is concerned with what is done with the data.

The remote services infrastructure 102 includes an infrastructure services portion 104 and an infrastructure communications portion 106. The infrastructure services portion 104 interacts with the plurality of service modules 103, as described in greater detail below. The remote services infrastructure 102 provides a set of application program interfaces (API's) that are used by a service module developer to leverage common services of the infrastructure such as database access, software delivery and notification services. The infrastructure communications portion 106 includes a plurality of communications modules 110.

The infrastructure services portion 104 interacts with a plurality of service modules 103. Examples of service modules that the remote services architecture may include are an administration and notification interface module 120, an installation, registration and change management module 122, an integration into system management platforms module 124, an integration into existing business systems module 126 and an API's for service module creation module 128. The administration and notification interface 120 allows a customer and service provider to control the remote services infrastructure. The installation, registration and change management module 122 supports the infrastructure and service modules deployed on top of the infrastructure. The module 122 may include automatic registration of new

software components, delivery of software and detection of changes within an environment. The integration into systems management platforms module 124 provides an integration point to systems management platforms in general. The integration into existing business systems module 126 allows the remote services infrastructure 102 to integrate into existing business systems to leverage data, processing capacities, knowledge and operational process. The module 126 allows the infrastructure 102 to integrate into the required business systems and provides interfaces to the service module creator to use those systems. The API's for service module creation module 128 allows a service module creator to abstract the complexity of remote data management. The module 128 provides an API of abstracted services to the service module creator.

The infrastructure communications portion 106 provides an abstraction of different protocol and physical network options. Examples of protocol options include an HTTP protocol and an email protocol. Examples of physical network options include Internet based communications, private network based communications and fax communications. The different protocol and physical network options are provided to meet the needs of as many customers as possible.

The infrastructure communications portion 106 supports a number of plug-in communications modules 110. Examples of the communications modules 110 include a communications authentication module 130, an encryption module 132, a queuing module 134, and a prioritization module 136. The communications authentication module 130 is related to the communications protocol that is used and provides the customer with authentication of a communication session. The encryption module 132 is related to the protocol being used and provides encryption of the data stream. The queuing module 134 provides the ability of the infrastructure to queue data being sent through the infrastructure to provide data communications integrity. The prioritization module 136 provides the ability for data within the system to be prioritized for delivery.

Referring to Figure 2, the remote services infrastructure architecture 205 includes a plurality of components. More specifically, the remote services infrastructure architecture 205 includes a remote services proxy 210, a remote services system management integrator 212, a remote services communications module 214, an intermediate mid level manager (MLM) 216 (which may be a customer MLM or an aggregation MLM), an applications MLM 218, a certificate

management system 220, a bandwidth management system 222, a remote services content generation MLM 224, a remote services application server 226. The remote services infrastructure architecture 205 interacts with a plurality of external service modules 103.

The remote services proxy 210 provides an API to the systems management systems. This API supports data normalization to the remote services data format. The remote services proxy 210 also provides receptors for the communications modules and in turn provides communications flow management using queuing. The remote services proxy 210 also manages allocation of remote services identifiers (ID's), which are allocated to each component of the remote services infrastructure, and the support instances that are registered with the remote services system 100.

The remote services system management integrators 212 are written to a remote services integrator API supported by the remote services proxy 210. One remote services proxy 210 can support many integrators (also referred to as integration modules). The integration modules provide the glue between the remote services system 100 and the systems management platform. There is at least one integration module for each support systems management platform.

The remote services communications modules 214 provide protocol, encryption and communications authentication. These modules plug-in through a semi-private interface into the remote services proxy 210, the intermediate MLM 216 and the remote services application MLM 218.

The intermediate MLM 216 may be either a customer MLM or an aggregation MLM. The remote services customer MLM is an optional deployable component. The remote services customer MLM provides a higher level of assurance to the customer-deployed environment, providing transaction integrity, redundancy and data queue management. The remote services customer MLM also provides an extensible environment through an API where service module components can be deployed. When no customer MLM is deployed, the aggregation MLM, hosted by the remote services provider and handling multiple customers, provides the data queue management, transaction integrity and redundancy. While the customer MLM is very similar to an aggregation MLM, a customer MLM may be required by a service module that needs to be localized. An aggregation MLM, being shared by multiple customers, may not be customizable.

The applications MLM 218 provides a series of functions that can exist on different MLM instantiations as applicable. The applications module provides data normalization, integration with the mail server data flow and integration with the certificate management system 220. This module acts as the gateway to the remote services application server 226 and controls data access.

The certificate management system 220 provides management of certificates to verify connection authentication for the remote services system 100. The certificate management system 220 may be horizontally scaled as necessary to meet the load or performance needs of the remote services system 100.

The bandwidth management system 222 provides control over bandwidth usage and data prioritization. The bandwidth management system 222 may be horizontally scaled as necessary to meet the load or performance needs of the remote services system 100.

The remote services content generation MLM 224 provides HTML content based on the data held within the remote services application server 226. This module provides a high level of HTML caching to reduce the hit rate on the application server for data. Accordingly, visualization of the data is done through the content generation MLM 224. Separating the visualization processing in the content generation MLM 224 from the data processing in the applications server 226 provides two separate scale points.

The remote services application server 226 provides the persistent storage of remote services infrastructure information. The application server 226 also provides the data processing logic on the remote services infrastructure information as well as support for the service module API to create service module processing within the application server 226. The application server 226 provides access to directory services which support among other things, IP name lookup for private network IP management. The application server 226 also provides access to the service modules 103.

In operation, the remote services proxy 210 uses the communication module 214 to connect to the intermediate MLM 216, whether the intermediate MLM is a customer MLM or an aggregation MLM. The applications MLM 218 and the intermediate MLM 216 use the certificate management system 220 to validate connections from customers. Dataflow bandwidth between the intermediate MLM 216 and the applications MLM 218 is controlled by the bandwidth management

system 222. Data that has been formatted by the applications MLM 218 is sent on to the application server 226 for processing and persistent storage.

The content generation MLM 224 provides visualization and content creation for users of the remote services system 100. Remote services infrastructure administration portal logic is deployed to the content generation MLM 224 to provide users of the remote services system 100 with the ability to manage the remote services system 100.

All of the remote services components are identified by a unique remote services identifier (ID). A unique customer remote services ID is generated at customer registration. For remote services infrastructure components, remote services IDs are generated, based on the customer remote services ID, at a component registration phase. For remote services entities reporting to a remote services proxy 210, such as a support instance or an integration module, the remote services ID is allocated by the proxy 210 itself, based on the remote services ID of the proxy 210.

Within the remote services architecture, there are instances where detection, collection and management logic (also referred to as systems management logic) may have already been created by another service module. In this instance, the service module creator reuses this functionality. The reuse then creates a more complex relationship within the system to be managed. The segmentation and re-use of data is available within the architecture. Instrumentation is made up of a large number of small data types. These data types are shared by the different service modules 103 using a publish and subscribe model.

In a publish and subscribe model, the remote services proxies (and therefore the systems management systems) publish their data to a service provider. The service modules 103 register interest in specific types of data that are needed to fulfill the respective service module processing. Figure 3 provides an example of the publish and subscribe model using example data and services.

More specifically, data from a systems management instrumentation proxy 306 may include patch information, operating system package information, disk configuration information, system configuration information, system alarms information, storage alarms information and performance information. This information is published via, e.g., a wide area network (WAN) to a management tier 310. Various service modules 103 then subscribe to the information in which they are respectively interested. For example, a patch management service module 330 might

be interested in, and thus subscribe to, patch information and operating system package information. A configuration management service module 332 might be interested in, and thus subscribe to, the disk configuration information, the patch information, the operating system package information and the system configuration information. A storage monitoring service module 334 might be interested in, and thus subscribe to, disk configuration information and storage alarms information.

Thus, with a publish and subscribe model, many different types of data are published by a customer using the remote services customer deployed infrastructure. Service modules then subscribe to these data types. More than one service module 103 can subscribe to the same data. By constructing the instrumentation data in a well segmented manner, the data can be shared across many services.

Sharing data across many services reduces duplication of instrumentation. By making data available to newly developed service modules, those service modules need to only identify instrumentation that does not exist and reuse and potentially improve existing instrumentation. Sharing data across multiple services also reduces load on customer systems. Removing the duplication reduces the processing load on the customer's systems. Sharing data across multiple services also reduces development time of service modules 103. As more instrumentation is created and refined, service modules 103 reuse the data collected and may focus on developing intelligent knowledge based analysis systems to make use of the data.

Accordingly, the separation and segmentation of the infrastructure from the service modules enables services to be created in a standardized manner ultimately providing greater value to the customer.

Referring to Figure 4, the remote services architecture includes a remote services API 402 which may be conceptualized in two areas, systems management API's 410 and remote services infrastructure API's 412.

The systems management API's 410 includes systems management API's 418, integrator 212 and proxy integrators API 430. The proxy integrator API 430 interfaces with integrator module service logic. The integrator module service logic is a general term for the configuration rules that are imparted on the systems management system to collect or detect the information for the integrator 212. While the proxy integrator API's 430 are not technically a part of the remote services system 100, the proxy integrator API 430 is used within the integration modules which form the boundary between the remote services system 100 and the system management.

The integration module creator provides the instrumentation to fulfill the collection and detection needs of the service via the systems management API 418.

The proxy integrators API 430 provides an interface between the systems management system and the remote services infrastructure 102. This interface provides a normalization point where data is normalized from the system management representation to a remote services standard. By normalizing the data, the remote services system 100 may manage similar data from different systems management systems in the same way. The proxy integrators API 430 interfaces with the remote services proxy 210 as well as the systems management integrator 212.

The remote services infrastructure API's are used by a service module creator and the systems management integrator 212. The remote services infrastructure API's 412 include an intermediate MLM Service Module API 432, an applications MLM API 434 and an applications server service module API 436 as well as a content generation MLM service module API 438. These API's provide the interface with the remote services infrastructure 102.

The intermediate MLM Service Module API 432 describes a distributed component of the infrastructure. The intermediate MLM service module API 432 allows modules to be loaded into this distributed component that provides mid data stream services such as data aggregation, filtering, etc. The intermediate MLM service module API 432 provides access and control over the data that flows through the intermediate MLM 216 to the service module provider. The intermediate MLM service module API 432 allows intercept of data upstream and on the back-channel to mutation, action and potential blocking by the service modules 103. The intermediate MLM service module API 432 interfaces with a service module creator as well as with the intermediate MLM 216 and intermediate MLM based service modules.

The applications MLM API 434 allows additional modules to be loaded on the applications MLMs. The applications MLM API 424 allows modules to be built into the applications MLMs 218 such as data normalization. The applications MLM API 424 interfaces with the applications MLMs 218 and modules within the applications MLM 218.

The applications server service module API 436 provides all of the needs of a data processing service module. The applications server service module API 436 provides access to many functions including data collected through a database and access to a full authorization schema. The applications service module API 436 is

based around the J2EE API. The applications service module API 436 provides a rich interface for service module creators to interact with and build services based on Enterprise Java Beans (EJB's) and data available to them. The application server service module API 436 interfaces with the remote services application server 226 and the service modules 103.

The content generation MLM API 438 is based around the J2EE web container and provides the service module creator a way of building a browser based presentation. The content generation API 428 interfaces with the content generation MLM 224 as well as with MLM generation based service modules.

The remote services infrastructure API's 412 also include a plurality of communication interfaces which are based around the extendibility of the remote services communications system. The communication interfaces include a communication protocol module 440, a communication encryption module 442 and an MLM infrastructure services portion 444. The communications interfaces interface with the remote services proxy 210 as well as all of the remote services system MLM's. The communications interfaces provide an interface between the communications modules and the components that use the communications modules.

The communications protocol module 440 provides support of the application level protocol that is used for the communication through the system. Modules of this type interface to support the use of Email and HTTP communications protocols. The communication protocol module 440 interfaces with remote services communications engineering personnel.

The communications encryption module 442 supports plug-in encryption modules. The plug-in encryption modules can either provide encryption at the protocol level or encryption of the data within the protocol. The communication encryption module 442 interfaces with remote services communications engineering personnel.

The MLM infrastructure services portion 444 represent a number of services that are included within the MLM that provide services that are relevant to the infrastructure 102. These services manage and manipulate the data as it passes through the different parts of the architecture. These services, such as queuing, utilize an API to access and manipulate the API.

Figures 5A and 5B show a more detailed block diagram of the remote services architecture depicted in Figure 2. Within this more detailed block diagram, the

remote services communications modules 214 are shown distributed across the remote services proxy 210, the intermediate MLM 214 and the applications MLM 218.

The remote services proxy 210 includes a remote services proxy foundation module 510 which is coupled to a communications module 214 as well as to a remote services proxy integrator API module 430, a remote services proxy ID management module 514 and a remote services proxy queuing module 516.

The remote services system management integrator 212 includes a systems management API 418 and a remote services integrator 212. The remote services integrator 212 is coupled to the remote services proxy integrators API module 430 of the remote services proxy 210.

Each communication module 214 includes a communications protocol module 520 and a communications crypto module 522. A communications module 214 may also include a communications authentication module 524.

The intermediate MLM 216 includes an intermediate remote services MLM foundation module 540 which is coupled between communication modules 214. The intermediate remote services MLM foundation module 540 is also coupled to a MLM queue and connection management module 542 and an intermediate service module API module 432. Communications modules 214 couple the intermediate MLM 216 to the remote services proxy 210 and the applications MLM 218.

Bandwidth management system 222 controls bandwidth usage and data prioritization on the communications between intermediate MLM 216 and applications MLM 218. Certificate management system 220 is coupled between the communications authentication modules 524 for the intermediate MLM communications module 214 and the applications MLM 218 communications module 214.

The applications MLM 218 includes a remote services MLM foundation module 550 that is coupled to the communications module 214 for the applications MLM 218. The remote services MLM foundation module 550 is also coupled to an MLM queue and connection management module 552 and the applications MLM API module 434 as well as a web server application server plug-in module 554.

Content generation MLM 224 includes a composition MLM foundation module 560. The composition MLM foundation module 560 is coupled to a service content generation module API module 438 and a remote services administration portal 564 as well as a web server application server plug-in module 566.

Remote services application server 226 includes an application server module 570 coupled to an application server service module API 436 and an infrastructure data management module 574. The application server module 570 is also coupled to relational database management system (RDBMS) 576. The infrastructure data management module 574 is coupled to a directory services module 578. The directory services module 578 is coupled to a data authorization system module 580 and user authentication modules 582. The user authentication modules 582 are coupled to human resources (HR) authentication module 590. The remote services application server 226 is coupled to a plurality of external service modules 230.

Figures 6, 7, 8, 9 and 10 show expanded views of the remote services proxy 210 and remote services system management integrator 212, intermediate MLM 216, applications MLM 218, applications server 226 and content generation MLM 224, respectively.

Figure 6 shows a block diagram of the remote services proxy 210 and the remote services system management integrator 212. The block diagram shows the delineation between the systems management software and the remote services system components as indicated by line 610.

The remote services proxy 210 provides an API via remote services proxy integrators API 430 which communicates using the operating system's Inter-Process Communication (IPC) implementation with the remote services proxy foundation module 510. This communication allows the API to be implemented with a number of different languages to meet the needs of the systems management developers while leaving a single native implementation of the remote services proxy foundation module 510. Examples of the languages used for the API include Java and C++.

The remote services proxy foundation module 510, together with the API 430, manage data normalization tasks. This ensures that systems management data is carried independently through the system. For example, an event from one type of service, such as a SunMC service, would have the same structure as an event from another type of service, such as the RASAgent service. Accordingly, the service modules may deal with the data types that are specific to the respective service and are independent of their source.

In the remote services architecture, the integrator 212 and proxy 210 are represented by two separate processes (e.g., address spaces). By representing the

integrator 212 and the proxy 210 as two separate processes, a faulty integrator 212 is prevented from taking down the whole proxy 210.

The remote services proxy queuing module 516 allows data to be queued for transmission when communications to the intermediate MLM(s) 216 become unavailable. This queuing is lightweight and efficient which in turn reduces the capabilities of length of time data can be queued and of reconnection management. The remote services proxy queuing module 516 provides a number of features that can be used to manage the queue, such as priority and time for data to live.

The remote services proxy ID management module 514 manages the allocation of unique identifiers for the proxy 210 itself and any support instances that are registered through the API. The remote services system 100 relies on the creation of unique ID's to manage individual support instances. This function is provided within the proxy 210 because there is no unique cross platform identifier available within the remote services system 100. The proxy 210 manages the mapping between the systems management ID (e.g., IP address) and the remote services ID, which is keyed off the unique customer ID provided at installation time within the deployed system.

Figure 7 shows a block diagram of the remote services intermediate MLM 216. The intermediate MLM may be a customer MLM or an aggregation MLM.

The customer MLM is an optional component that can be deployed to support scaling of both support instances and services as well as provide enhanced availability features for a deployed remote services environment. The intermediate MLM 216 receives information via the HTTP protocol from the remote services proxy 210. This information may optionally be encrypted. Connections are not authenticated by default on the server side, as it is assumed that the connection between the intermediate MLM 216 and the proxy 210 is secure.

The intermediate remote services MLM foundation module 540 exposes the data flow to the service module API 432 where registered service modules can listen for new data of specific types and mutate the data as required. Examples of this function include filtering of certain types of data or data aggregation. The customer MLM does not keep state from an infrastructure perspective. However, the service module could choose to keep persistent state information. The recoverability fail-over support of that state, however, is in the domain of the service module, although the

basic session replication features that provide the redundancy features of the infrastructure data flow may be reused.

The queue and connection management module 542 provides a highly reliable secure connection across the wide area network to the service provider based MLM farms. The queue manager portion of module 542 also manages back-channel data that may be intended for specific remote services proxies as well as for the applications MLM 218 itself.

The intermediate remote services MLM foundation module 540 manages the rest of the MLM's roles such as session management, fail-over management and shared queuing for the back-channel.

Aggregation MLM's, while provided by the service provider, function much the same as customer MLM's. Strong security is turned on by default between such MLM's and the remote services proxy 210. Accordingly, a communications authentication module 524 is used on the receiving portion of the intermediate MLM 216.

Referring to Figure 8, the remote services application MLM 218 provides several functions (applications) for the remote services system 100. The remote services application 218 hosts applications as well as functioning as a content creation MLM. The host applications within the application MLM 218 include data normalization, customer queue management and remote access proxy. The data normalization application supports normalization and formatting of data being sent to the application server 226. The customer queue management application handles general connections to and from customer remote services deployments. The customer queue management application also manages back-channel requests and incoming request. The remote access proxy application provides a remote access point as well as functioning as a shared shell rendezvous point. The applications MLM 218 uses the application server plug-in to communicate directly with the application server 226.

The communications authentication module 554 communicates with the certification management system 220 to validate incoming connections from customers. Each customer is provided a certificate by default although more granular allocations are available. Certificates are distributed at installation time as part of the installation package for both the remoter services proxy module and for the remoter services customer MLM.

Referring to Figure 9, the application server 226 manages the persistence and data processing of the remote services infrastructure 102 and the service modules 103.

The application server 226 provides the core service module API 436 to the service module creator. The service module API 436 is based upon the J2EE API. The service module API 436 allows the service module creator to register for certain types of data as the data arrives and is instantiated. This data can then be processed using the support of the application server 226 or alternatively exported from the remote services system 100 for external processing.

The infrastructure data is held within the application server 226 and stored within the RDBMS 576 associated with the application server 226. Access to this data is available via the service module API 436 and is managed via the infrastructure data management module 574.

The directory services implementation supports user authentication, data authorization and private network data support. User authentication uses a pluggable authentication module (PAM) so support a plurality of authentication methods such as a lightweight directory assistance protocol (LDAP) method for service provider employees and a local login method for a remote services based login schema. Other methods may be added. The LDAP login is processed using a replicated copy of an LDAP server running within the remote services infrastructure 102.

Data authorization is designed to protect the data held within the application server 226 to specific groups of users. This protection allows customers to grant or deny access to their service data to specific users. This data protection is managed down to the service module granularity. So for example, a customer could grant information about advanced monitoring on a subset of their support instances to members of a service provider monitoring staff.

Referring to Figure 10, the remote services content generation MLM 224 provides HTML generation bases on the data held within the application server 226. The content generation MLM 224 provides a service module API 438 for service module creators to develop content composition for their data which is processed by the application server 226. The content is in the form of J2EE web container which supports Java servlets and Java servlet pages (JSP) API's.

The content generation MLM 224 communicates with the application server 226 using the same Netscape API (NSAPI) plug-in as the remote services applications MLM 218. Instances of these two MLMs make up an MLM farm. The composition

remote services foundation layer provides support for caching of HTML pages and associated data to reduce the data request hit back to the application server 226.

The remote services administration portal 564 provides control of the deployed customer infrastructure to the customer and control over the total infrastructure to trusted users.

Figure 11 shows a flow diagram of communications within a remote services architecture. In one embodiment, the communications between a customer and a service provider is via a wide area network (WAN). Communications within the remote service architecture includes three tiers, a remote services proxy tier 1110, an intermediate MLM tier 1112 and an application MLM and server tier 1114. Communication is established and connections are made from the bottom tier (the remote services proxy tier) to the top tier.

The remote services architecture supports two application protocols for the majority of its services classification support: HTTP and Email messaging. There are a plurality of service module classifications that each have specific communications protocol relationships. More specifically, the service module classifications include a data collection classification, a monitoring classification, a remote access classification and an infrastructure administration classification.

With the data collection classification, the connection orientation is message based, the physical connection support may be Internet, private network or fax, and the protocols supported may be Email or HTTP. Examples of service modules of this classification include an inventory management service module and a performance management service module.

With the monitoring classification, the connection orientation is message based, the physical connection support may be Internet, private network or fax, and the protocols supported may be Email or HTTP. Examples of service modules of this classification include basic self service monitoring and full hardware monitoring with service action.

With the remote access classification, the connection orientation is session based, the physical connection support may be Internet, private network or fax, and the protocol supported is HTTP. The session based connection orientation is one way initiation from the customer. Examples of service modules of this classification include remote dial in analysis and remote core file analysis.

With the infrastructure administration classification, the connection orientation is session based or off-line installation, the physical connection support may be Internet, private network or fax, and the protocol supported includes HTTP, email or physical (e.g., telephone or CD). The session based connection orientation is one way initiation from the customer and the off-line installation is via, e.g., a CD. Examples of service modules of this classification include remote services administration, installation, updates, configuration and notification.

Encryption options are related to the protocol. A secure socket layer (SSL) protocol, for example, is likely to be the chosen protocol for an HTTP transmission, i.e., an HTTPS transmission. The remote services communication architecture does not enforce this however. So, for example, data could be sent by encrypting the body of an HTTP stream. This provides an advantage when a customer's HTTPS proxy infrastructure is not as resilient as their HTTP proxy infrastructure.

Email uses an email encryption option such as s-mime or encrypting the body using a third party encryption method such as PGP. Encryption is optional at all stages. If the customer does not require encryption, then encryption need not be used.

Authentication of the remote services communication is standard for all protocols. Accordingly, the service provider may validate the sender of data and the customer may validate that the service provider is the receiver. Authentication is managed via certificates.

Certificates are used in both the client and server to authenticate a communications session. Client certificates are generated during the customer registration process and are built into the remote services proxy and the customer MLM. By default, each customer is provided a client certificate. The customer can, however, define specific security groups within their service domain and request additional client certificates for those domains. Remote services processes include a certificate distribution mechanism, supporting either the creation of a new security group within an existing customer or the redeployment of a new certificate after a certificate is compromised.

Figure 12 shows a block diagram of the data blocks that comprise the data that flows through the remote services infrastructure. Each system management system conforms to the data definitions that are part of the remote services proxy integrators API 430. The remote services communications architecture provides a normalized

view of the data, regardless of in which systems management framework the data originated.

Data block header 1202 is common to all data types. Data block header 1202 contains items such as source, routing information, time to transmit and source type. Data block header 1202 is used to route the data correctly through the remote services system 100 to the correct service module 103. Data block header 1202 is used to provide diagnostic and quality of service measurement built into the system.

Infrastructure data block 1204 provides data classification service classification specific data. Infrastructure data block 1204 removes systems management specific data.

Service module data block 1206 provides format based on each service classification that drives the system the systems management normalization of the data that flows through the system. For example, alarm data includes general characteristics defined such as severity, state and originating support instance.

Figures 13A and 13B show an example of the component relationships of a remote services system 100 that is configured according to the remote services architecture. Various components of the remote services system 100 execute modules of the remote services infrastructure architecture 205. Remote services system 100 includes customer deployment portion 1302a, 1302b, network portion 1304, data access portal 1306a, 1306b, Mid Level Manager (MLM) portion 1308, and application server portion 309.

Customer deployment portion 1302a sets forth an example customer deployment. More specifically, customer deployment portion 1302a includes SunMC server 1310, WBEM agent 1312, and Netconnect Agent 1314. SunMC agents 1316a, 1316b are coupled to SunMC server 1310. Server 1310, Agent 1312 and Agent 1314 are each coupled to a respective remote services proxy 1320a, 1320b, 1320c. Remote services proxies 1320a, 1320b, 1320c are coupled to network portion 1304, either directly, as shown with proxy 1320c, or via customer MLM 1322, as shown with proxies 1320a and 1320b. Proxies 1320a and 1320b may also be directly coupled to network portion 304 without the MLM 1322 present. The SunMC server is a provider specific systems management server (i.e., health management server). The SunMC agents are provider specific systems management agents (i.e., health management agents). The WEBM agent is a web based enterprise management agent. The Netconnect agent is a basic collection agent. Customer deployment portion 1302a

illustrates that the systems management may be 2-tier (e.g., agent, console) or 3-tier (e.g., agent, server, console).

Customer deployment portion 1302b sets forth another example customer deployment. More specifically, customer deployment portion 1302b includes RasAgent 1330, SunMC agent 1332, NS server 1334 and Netconnect Agent 1336. RasAgent 1340 is coupled to RasAgent 1330. SunMC Agent 1342 is coupled to SunMC Agent 1332. NSAgent 1344 is coupled to Netconnect Agent 1336. RasAgent 1330 and SunMC Agent 1332 are coupled to remote services proxy 1350a. Metropolis Server 1334 is coupled to remote service proxy 1350b. Netconnect Agent 1336 is coupled to remote services proxy 1350c. Remote services proxies 1350a, 1350b, 1350c are coupled to network portion 1304 either via customer MLM 1352 or directly. The RasAgent is a reliability, availability, serviceability agent. The NSagent is a network storage agent and the NS server is a network storage server. Both the NSagent and the NS server are reliability, availability, serviceability type devices.

Network portion 1304 includes at least one interconnection network such as the Internet 1354 and/or a private dedicated network 1355. Internet 1354 is assumed to be an existing connection that is reused by the remote services system. The private dedicated network 1355 is a dedicated link that is used exclusively by the remote services system to connect the customer to the service provider. The data to manage the private network is provided by directory services technology held within the application server portion 1308. The directory services technology handles all of the domain name service (DNS) services used to manage name to allocated internet protocol (IP) information. The remote services infrastructure also offers transmission over fax from the customer's environment (not shown). The fax communication is for service modules where the fax transmission makes sense. For example, fax transmission may be used in a military site which does not allow electronic information to be transmitted from it.

Data access portal portions 1306a and 1306b provide access to the remote services system 100. More specifically, data access portal portion 1306a includes a service access portion 1360, a customer access portion 1362 and a field information appliance (FIA) 1364. Data access portal portion 1306b includes a partner access portion 1366 and a system management interface (SMI) data access portion 1368.

Mid level manager portion 1308 includes load balancers 1370a, 1370b, MLM webservers 1372a, 1372b, 1372c and communication authentication (CA) and de-encryption server 1374.

Application server portion 1309 includes a plurality of application servers 1380a – 1380f. Application servers 1380a, 1380b are associated with transactional and infrastructure data storage 1384a. Application servers 1380c, 1380d are associated with transactional and infrastructure data storage 1384b. Application servers 1380e, 1380f are associated with transactional and infrastructure data storage 1384c. Application server portion 1309 also includes knowledge base 1390a, 1390b. Application server portion 1309 integrates with service applications as well as via generic data export (such as, e.g., XML).

Remote services proxies 1320, 1350 provide a System Management Integrators API. Using this API, system management products can integrate their customized handling of data into the common data format that is used by the remote services architecture. Accordingly, the system management component of the overall system is effectively segmented away from the remote services architecture.

Additionally, by using the remote services proxies 1320, 1350, the remote services architecture leverages much of a pre-existing instrumentation and data collection mechanisms that already exist. Accordingly, already deployed instrumentation agents within a remote service provider existing system such as those from SunMC and Netconnect may be integrated into a remote services system. Additionally, third party systems management systems may also be supported and integrated via the remote services proxies.

Customer deployment portions 1302a, 1302b each show an optional customer MLM component deployed to the customers environment. Whether to deploy the customer MLM component depends on a number of factors. More specifically, one factor is the number of support instances installed in the customer's environment and the number of services being utilized by the customer. A deployed MLM component can allow greater scale capabilities. Another factor is the type of services deployed within the customer environment. Some services are optimized when an MLM component is deployed to the customer environment to support service specific tasks such as filtering and data aggregation. Another factor is the quality of service. Deploying an MLM component provides a greater level of quality of service because

the MLM component provides enhanced data communications technology within the MLM infrastructure modules.

The decision of whether to deploy a remote services MLM component (or more) to the customer's environment is a deployment decision. There are a number of architecture deployment classes which are used to meet the varying customer needs.

The remote services system communicates via two main protocols, HTTP and email. Security considerations for these protocols can be chosen by the customer and plugged into the system. For example, the HTTP protocol may use SSL. Additionally, the email protocol may use some well known form of encryption.

The connections from the customer deployment portion 1302 feed into MLM farms which reside within the SMI service provide environment. These MLM farms are sets of redundant web servers 1372 that are balanced using conventional load balancing technologies. Alongside these web servers 1372 are infrastructure servers 1374 which provide specific infrastructure acceleration for decryption and distribution of certificates for communications authentication.

These MLM farms provide a plurality of functions. The MLM server farms provide remote proxy connections. In deployments when an MLM is not deployed to the customer, the customer's proxy connects to the MLM farms within MLM portion 1308. Also, in deployments when a customer MLM 1322, 1372 is present, the MLM farm communicates and manages communication with the deployed customer MLM 1322, 1372. Also, the MLM server farms provide data processing capabilities, e.g., the MLM farms provide application specific tasks to prepare data for passing to the remote services application server portion 1309. Also, the MLM server farms provide access points for the customer and service personnel via browser like connections. The MLM farm generates the HTML that is presented to the browser.

The MLM technology is based upon known web server technology such as that available from Sun Microsystems under the trade designation iPlanet. Plug-in functionality is provided by the servlet and JSP interfaces available as part of the web server technology.

The remote services application servers 1380 provide data processing and storage for the remote services infrastructure as well as for any hosted service modules. The remote services application servers 1380 are based upon known application server technology such as that available from Sun Microsystems under the

trade designation iPlanet application server 6.0. The remote services application server 1380 provides support for horizontal scalability, redundancy and load balancing. Thus providing the back-end components of the remote services architecture with a high level of built in assurance and flexibility. Application partitioning of the application servers 1380 provides processing distribution to ensure that heavy processing that may be required by more complex services are handled appropriately without affecting the remainder of the remote services architecture.

Application server portion 1309 provides integration into existing business systems, generic data export and tight integration with existing knowledge base implementations 1390. Data export is handled through structured XML, data can be exported asynchronously by a client registering to receive data of a particular type or synchronously by the application server 1380 accepting a request from a client.

The core service module API is provided by the application server 1380 using a J2EE implement API. The basic container services of J2EE are extended to provide remote services specific functions and to create the basis of the API. Accordingly, a service module creator can rely on a number of provided for services, such as database persistency, high levels of atomic, consistent, isolated, and durable (ACID) properties, directory service access, authorization protection for the data and access to the data collected by the remote services infrastructure itself.

The creation of a service module, which provides the technology to support a specific remote service, involves at least one of the following components: a creation of detection/collection logic component; a mid-stream analysis and management of data component; an analysis and storage of data component; and, a presentation and management of the data/knowledge component.

The detection/collection logic is created within the domain of a systems management toolkit. The mid-stream analysis and management of data is an optional step and effectively provides analysis of the data within the customer's environment. Inclusion of this logic would mean that the mid-stream analysis and management of data service module would have a remote services MLM deployed to the customer's environment 1302a, 1302b. The deployment of the remote services MLM to the customer's environment reduces and manages the data being sent over the WAN to the remote services provider. The analysis and storage of data component is performed within the application servers domain (the component may be exported). This analysis and storage of data component turns data into knowledge and service

value that can then be presented back to the customer. The presentation and management of the data/knowledge component is where the data and knowledge that is developed from the analysis and storage of data component is presented to the customer or service personnel. The presentation and management of data/knowledge component may include interactive support to provide modification of the data values.

The remote service delivery system involves an infrastructure using a wide range of different technologies working together to collect and process data. The system, however, should be simple to deploy and administer. The deployment and maintenance of the system comprises two functions. The first component deals with the physical operation which is generally handled at the customer site. This component may involve installation of proxy 210 software or connecting an MLM appliance. It may also involve some configuration related to the above-mentioned components. These configuration parameters, however, are not related to the remote service delivery system 100 itself but, rather, to the external configuration.

The second function deals with the remote service delivery system 100 itself. Specifically, this component deals with the data model associated with the remote service delivery system. Most of the objects and relationships described in this data model symbolize the logical organization of a remote service delivery system deployment, with the exception of a few objects stored for disaster recovery purposes only. Through this data model, the customer is able to organize how the remote service delivery system is deployed. The data model shows communication subparameters, for example, which protocol is used, e.g., HTTP, or email; the data model also shows organization parameters, for example, which is the next MLM in the network; and the data model identifies security parameters such as certificates.

If changes to the data model impact existing components, the remote services system back-channel is used to push the changed configuration back. In some cases, however, the back-channel may be broken or nonexistent. In such cases, the system

100 provides a procedure to make reconfigurations happen in the most automatic manner possible.

In the remote services system 100, data configuration parameters are stored on servers as data objects that constitute the communication and identification parameters. These parameters are provided to the infrastructure components either upon request at the initial installation phase or are "pushed" to the components if changes are subsequently made to the data model.

As discussed above, one of the problems that is often encountered is that the communication path used to transmit the configuration information may be broken. To counteract this situation, the system provides an automatic procedure to restore connectivity and, thereby, to access the servers and the system parameters stored thereon. The automatic reconfiguration solution is accomplished as follows: If the system detects a communication error related to an identify problem, such as an invalid client certificate, the system communication module will try to fetch a new certificate from a secure URL pointing to the service provider web site containing data parameters relating to the system. Customer and component information such as the customer ID and the component ID are associated with this request via a HTTP POST. This request is redirected to a web server local to the customer and will reply with the new certificate which is installed automatically.

If the system detects a communication problem not related to identify (cannot reach the MLM, for example), it will then try to fetch its configuration from a secure URL pointing to the service provider web site. Certificate information, such as the customer ID and the component ID, are associated with this request via a HTTP POST. However, this request may be served directly by service provider servers or redirected to be served by a web server local to the customer network.

When a component receives its new configuration under either of the scenarios discussed above, from a local web server or from the service provider, it will install it and validate the connectivity back to the system servers. The procedure for detecting errors and automatically reconfiguring the system is discussed in greater detail below in connection with the data processing flowcharts.

This use of redirect, which may be generated by the application server 226 or by a customer owned HTTP proxy located on the path to the application server 226, enables a centralized way to distribute configuration files to the customer network and improve automatic installation and re-configuration for complex system deployment.

Prior to discussing automatic reconfiguration, it is useful to summarize the steps taken by a customer to provide an automatic configuration of this site during the installation phase. At the installation phase, the customer creates a new system component, such as a Proxy 210 or Customer MLM, which requires settings or configuration parameters. Since most of these settings exist in the remote services system database, these settings can be provided to the new component automatically.

Configuration information is used in two different cases: First, configuration information is needed during the installation phase to provide a new component with its generic settings and to enable it to communicate with the remote services system servers. In this case, the configuration information used is generic to all objects which belong to the same MLM or Proxy group. The information specific to this component is gathered during the installation and stored. Second, configuration information is needed to recover a component after a crash. In this case, the information is specific to the particular object affected by the crash. Installation is a common task, and should be as automatic as possible for the customer. Disaster

recovery is an exception case, which requires some manual action on the part of the customer.

Configuration information is provided to the remote services system component as an XML file, stored locally. Configuration, generic to a MLM or Proxy group, or specific to an existing Proxy or Customer MLM, can be extracted from the remote services system database through service provider web Administration Portal. The result is provided as an XML file.

To automate installation, the remote services system implements the following procedure: First, the system checks to determine whether a local configuration does exists in the installation directory. If the local configuration does exist, it is used to configure the system components. Second, if a predefined environment variable is set, the system attempts to fetch the URL defined in this variable. If the fetch is successful, the system uses that URL in conjunction with the variable definition. Finally, the system attempts to fetch the configuration using a SSL URL pointing to a service provider server.

The URL used to point to a configuration can be an HTTP URL (http://...), a SSL URL (https://...) or a file URL (file:///..). When a URL is an HTTP or SSL URL, the system will automatically connect to it, using the HTTP GET method, supplying some local parameters as the component IP address(es) and name.

Reconfiguration is triggered by the remote services system communication module upon detection of an error. When this error is due to either a rejection of the client certificate or a new connectivity problem, the remote services system communication module will try to fetch a new configuration or certificate and will attempt to re-validate the connection. This enables reconfiguration of a broken deployment with the minimal manual operation on the part of the customer.

Figure 14 provides a flowchart illustrating the processing steps implemented by the remote services system 100. In step 1400, the system detects that a documentation error has been detected and proceeds to step 1402 where the error condition is tested to determine if the client certificate has been rejected. If the test determines that the error is related to a rejected client certificate, the system proceeds to step 1404 to provide a new certificate to the client and the system proceeds to step 1410 discussed below. If the test performed in step 1402 determines that the detected error is not related to a rejected client certificate, the system proceeds to step 1406 where the error condition is tested to determine if it relates to a connectivity problem. If the result of the test in step 1406 indicates that the error is not related to a connectivity problem, the system proceeds to step 1414 where further testing is aborted. If, however, the test performed in step 1406 indicates that the error is related to a connectivity problems, the system proceeds to step 1408 and issues a new client certificate. In step 1410, a test is performed to ensure that there is a good connection with the remote services server. In step 1412, the result of the connection is again tested to detect any error conditions related to the server connection. If an error is detected, the system proceeds to step 1414 where further testing is aborted. If no error is detected, the system proceeds to step 1416 where a message is issued to indicate that the system is back in working order. Two sub-processes are used in this figure.

The first sub-process, described below in Figure 15, is implemented for the retrieval and installation of a new client certificate. In step 1500, the system 100 receives an error message indicating that the client certificate has been rejected. In step 1502 the system attempts to obtain a HTTP redirect and a test is performed in step 1504 to determine if the HTTP redirect was received. If the HTTP redirect was

not received, the system proceeds to step 1516 where an error message is issued. If the HTTP redirect was successfully received, the system proceeds to step 1506 where it attempts to "Get New URL." In step 1508 the New URL is tested and the system proceeds to step 1516 if an error is detected. If no error is detected, the system proceeds to step 1510 where a new client certificate is received and installed. In step 1512 the connection to the remote service server is tested and a determination is made in step 1514 regarding the existence of any errors in the connection. If errors in the connection are detected, processing proceeds to step 1516 to issue an error message. Otherwise, processing proceeds to step 1518 where a message is issued to indicate that the system is in working order.

Note that in the above-described reconfiguration process a HTTP redirect is required. The application servers 226 may generate the redirect upon a request from the customer or the request may be generated by the customer network infrastructure. Therefore certificates are never stored on a public server related to the remote services system. Instead, the customer publishes the certificate on its network, following its own security policy.

The second sub-process illustrated in Figure 16 is followed by the remote service system for reconfiguring the system component in response to a communication error. In step 1600, the system detects that a communication error exists and processing proceeds to step 1602 where the system attempts to get a HTTP redirect. In step 1603, a test is conducted to determine whether the HTTP redirect was received. If the test in step 1603 indicates that the HTTP redirect was not received processing proceeds to step 1606 where the error is confirmed and a HTTP redirect error message is issued in step 1614. If the test performed in step Z03 indicates that the HTTP redirect was successfully received, the system proceeds to

step 1604 to "Get New URL." In step 1606, the New URL is tested and if an error is detected a HTTP redirect error message is issued in step 1614. If no error is detected, processing proceeds to step 1608 where the new communication configuration is installed. In step 1610, the connection to the remote service servers is tested and any errors are then determined in step 1612. If an error is detected the system proceeds to step 1614 where a HTTP redirect error is issued. Otherwise the system proceeds to step 1616 where a message is issued indicating that the system is back in working order.

Note the difference in the certificate download process described in Figure 16. In this case, the application server 226 may reply directly to the first HTTP request by providing the communication parameters without sending a HTTP redirect. Both of the processes shown in Figures 15 and 16 use HTTP redirect as a way to distribute these configuration parameters to a network reachable by the broken system component. In the case of a connection configuration change, new parameters may be served directly by service provider server. Certificates, however, should be served by a customer web server.

The use of the certificates in the remote services system 100 is a very simplified use of a PKI infrastructure. Client certificate verification happens only on the service provider servers, which has direct connectivity to the PKI server through a private network. Thus, the process associated with certificate management is simplified.

As can be seen from the foregoing discussion, certificate distribution has been simplified at the origin by combining the client certificate within the software package. There are two cases, however, that need particular attention. The first case is when a customer contacts the service provider to ask for the revocation of its

certificate and the allocation of a new one. This case may happen if the customer certificate was compromised. To avoid spoofing of the customer identity, service provider should provide a way to revoke the customer's existing certificate, so that all service provider servers can reject spoofed information reaching them. At the same time the existing certificate is added to the revocation list, a new certificate is generated and should be distributed to the customer.

The second case happens when the customer wants to use multiple identities, through different security groups, representing different identities in this organization. Here, the service provider also generates a new certificate and distributes it to the customer.

To distribute a new certificate, service provider needs to ensure that the receiver is the real customer and not someone impersonating the customer by using its compromised certificate. For security reasons, the service provider distributes the new certificate only through a service provider web administration portal, requiring the customer username and password for access, or through a physical media sent to the customer address.

The installation of the certificate is performed by the service provider communications module 214 and triggered by a communication error stating an invalid certificate as described above. When such an error is received, the service provider communications module will try to download a secure URL pointing to a service provider server, associating to it its customer ID and the service provider ID of the caller. This URL, hosted by the service provider, will never return the certificate.

Two cases are possible: either the customer has a http-proxy or firewall on the path to the service provider, or not. In both cases, it is desirable for the reply from

this URL request to be a HTTP redirect pointing to a web site local to the customer. Thus, the customer can first download his new certificate to a local and private web site. Then, if the customer has a HTTP proxy or a firewall, he can intercept the request to the service provider certificate web server to redirect them to its private web server, or ask the service provider to setup his web server to perform the redirect. This mechanism was explained above in connection with reconfiguration. In both cases, the real identity of the customer is verified, the certificate is delivered to him in a secure way and the distribution of it stays on his private network.

Other Embodiments:

Other embodiments are within the following claims.